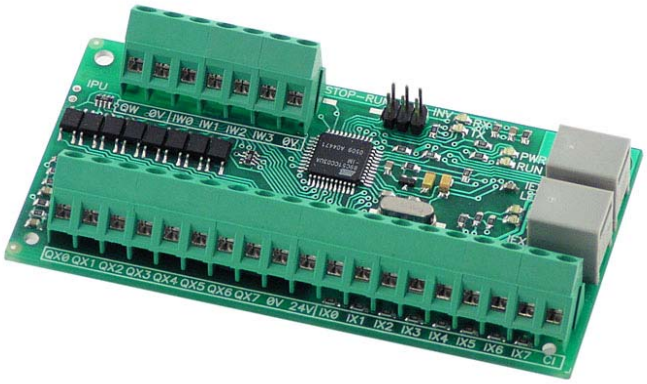
IPU User Manual

rev. 9
(applies to IPU IEX mode v1.4/PLC mode v2.3/CyPro 2.3.4 and later)



cybroTech

© 2006,2007 Cybrotech Ltd

Cybrotech Ltd
14 Brinell Way, Harfreys Industrial Estate
Great Yarmouth, Norfolk, Nr31 OLU - UK
tel: +44 1493 650 222
www.cybrotech.co.uk
info@cybrotech.co.uk

General description

IPU (Inverter Programming Unit) is a PLC for integration with Hitachi SJ300/L300P inverter.
It has:

- 8 digital inputs
- 8 digital outputs
- 4 analog inputs
- 1 analog output
- IEX-2 (CAN) port
- serial communication to inverter

IPU is placed into inverter option slot:

IEX-2 port may be used for connecting:

- CyBro-2
- operator panel
- other IPU-s

External, fully programmable operator panel is available.

Using IPU, inverters are capable of performing tasks not possible before, such as time-controlled operation or group-synchronize operation. Some application examples are listed here:

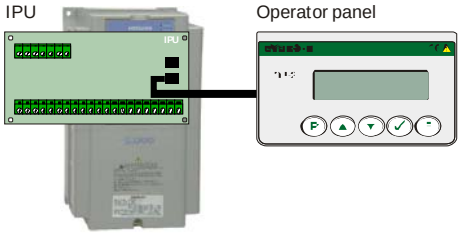
Parameter access	Access to selected application-specific parameters, with optional password protection.
Dosator machine	Precise dosator using hose-pump control and internal counter.
Enhanced multi-stage	Multi-stage with control logic and programmable timers.
Fail-safe pump station	Using intelligent monitoring, failure may be detected, allowing pump-station to continue normal operation.
Equalizing operating hours	Using intelligent power-management control, inverters take care that load is evenly distributed among pumps, providing longer pump life and longer service period.
Regulation with sleep-mode	If frequency is below predefined level, IPU may automatically put inverter into sleep mode.

Operation modes

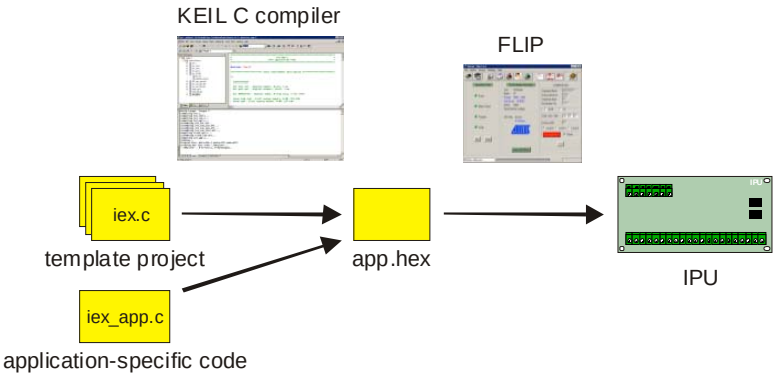
IPU may operate in two different modes:

1) PLC mode

IPU is a stand-alone controller, programmed according to application.



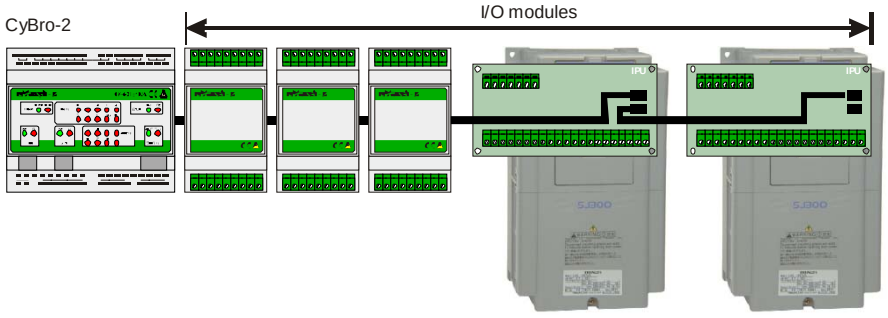
Programming is made in C, adding application logic to ready-made template. Input/output and communication are handled by the template. User has to put application-specific code into "iex_app.c" module and compile the project.



Program is loaded to IPU using ISP-adapter and FLIP software. For more details about loading, please check "ISP programming.pdf".

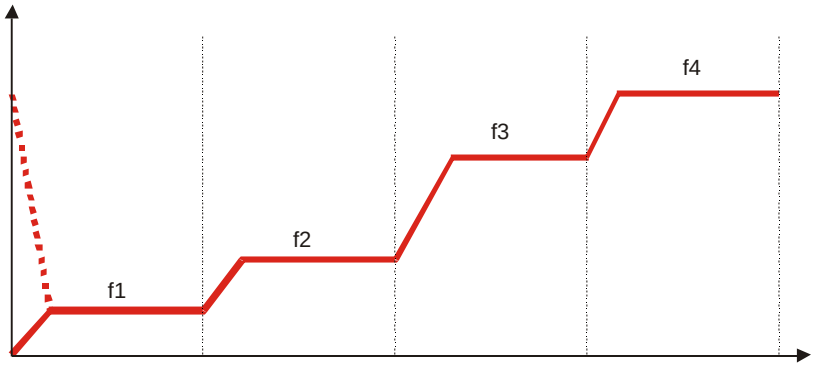
2) IEX-2 mode

IPU is CyBro-2 external i/o module. Program logic is implemented in CyBro, IPU is used to access inverter parameters.



PLC mode example

Demo application is an example of multi-speed operation.
Four setting frequencies may be edited by operating panel.

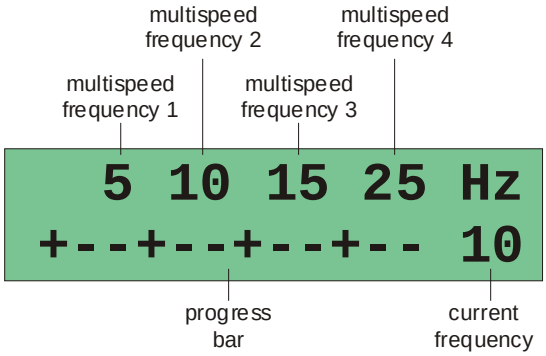


Predefined parameters:

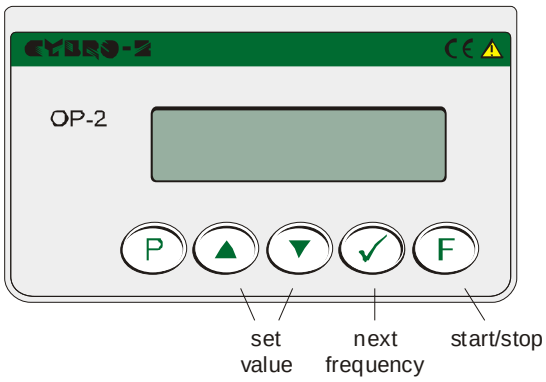
A001=2 (frequency source to keypad)
A002=2 (run command source to keypad)
F002=500 (acceleration time to 5s)
F003=500 (deceleration time to 5s)

Parameters are sent immediately after power-on.

Panel preview:



Command keys:



PLC mode programming reference

input/output
---
bit ix0..ix7 - digital inputs, 0-off, 1-on
bit qx0..qx7 - digital outputs, 0-off, 1-on
uchar iw0..iw3 - 8-bit analog inputs, 0-0V, 255-10V
uchar qw0 - 8-bit analog output, 0-0V, 255-10V
bit RUNSWITCH - run/stop switch, 0-stop (on), 1-run (off)
bit RUNLED - run LED, 0-on, 1-off

inverter parameters
---
struct ip[0..15] - input parameters
struct qp[0..15] - output parameters

ip/qp fields
---
uint address - parameter address
uchar size - parameter size in bytes (1..4)
uint value - parameter value
uchar event - request to read/write parameter from/to inverter

For input parameters, event should be set to desired read period:

0 - never
1 - every 1000ms
2 - every 100ms
3 - every 10ms

For output parameters, event is used as request to send parameter:

128 - request to send (defined as WRITE_PENDING)

timers
---
bit timer_10ms;
bit timer_100ms;
bit timer_1000ms;

Used by user application to perform time-synchronized tasks.
Bits are set by system at predefined intervals, and should be cleared by user application.

functions
---
bit _testbit_(bit x) - read bit and clear, return 1 if x is 1, set x to 0

operating panel
---
uchar disp_buf[32] - display buffer, starting from up left corner
bit refresh_display_req - request to send buffer to panel (set automatically for print functions)

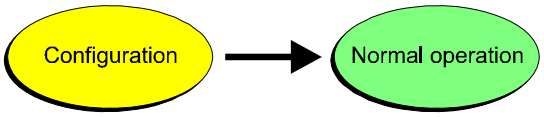
bit key_up,key_dn,key_e,key_f - current state of panel keys

bit send_empty_mask - used for panel initialization
uchar mask_data_count - used for panel initialization

void at(uchar x, y) - set cursor position
void print_string(char code *p) - print an ascii string on cursor position
void print_integer(long v, uchar n, uchar d, uchar zb) - print a long integer on cursor position
(v-value, n-places, d-decimals, zb-zero blanking)

IEX-2 mode operation

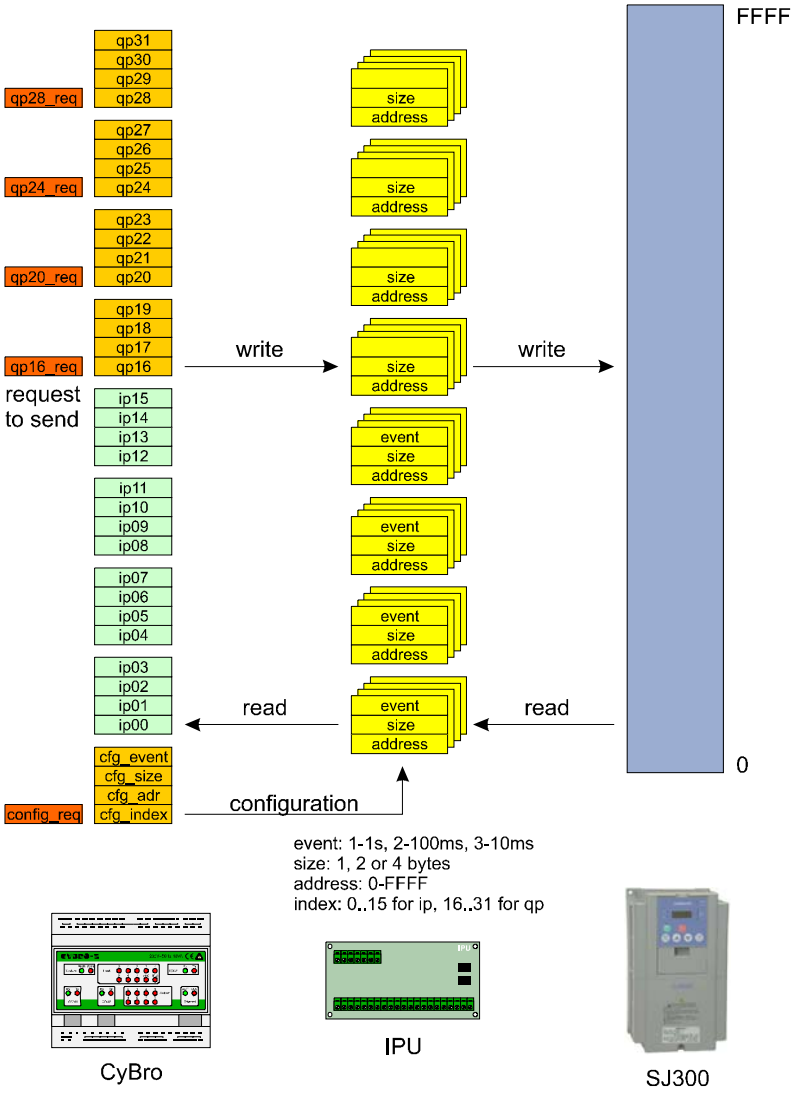
Before reading or writing inverter parameters, IPU should be configured. Configuration is a process of selecting parameters which will be used during the normal operation.



Two separate configuration areas are available, ip for reading and qp for writing parameters.

Both areas are divided into four blocks, which are sent and received independently. Each block contains four variables (ip00-ip03, ip04-ip07, ip08-ip11, ip12-ip15, qp16-qp19, qp20-qp23, qp24-qp27, qp28-qp31). Each variable may be configured separately (address and size). 3 or 4-byte inverter value is received in two variables.

Input variables should also have specified reading frequency (event). Event should be zero for output variables.



After configuration, read variables will be updated periodically. To write a parameter, write value into variable and set the request bit.

IEX-2 mode examples

1. Reading current inverter frequency

Parameter: D001
Address: $400
Size: 4 bytes
Update: 100ms

Configuration:

```
if first_scan then
   ipu00_config_index:=0; // read to ip00
   ipu00_config_address:=1024;
   ipu00_config_size:=4;
   ipu00_config_event:=2; // 100ms interval
   ipu00_config_req:=1; // send configuration
end_if;
```

Normal operation:

```
current_frequency:=65536*word(ipu00_ip00)+word(ipu00_ip01);
```

2. Run/stop inverter

Parameter: A020
Address: $472
Size: 1 byte
Update: on request

Configuration:

```
if first_scan then
   ipu00_config_index:=16; // write from qp16
   ipu00_config_address:=1138;
   ipu00_config_size:=1;
   ipu00_config_event:=0; // ignored
   ipu00_config_req:=1; // send configuration
end_if;
```

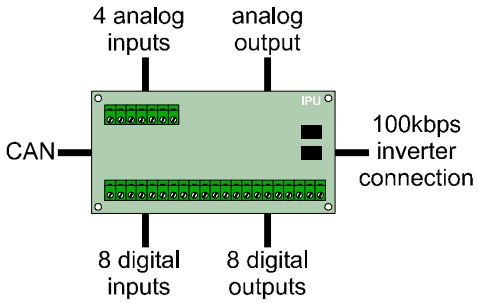
Normal operation:

```
ipu00_qp16:=1; // run
ipu00_qp16_req:=1; // request to send
```

Please also check "IPUDemo.cyp" from CyPro Examples directory.

Technical specifications

Picture below summarizes IPU inputs, outputs and communication channels:



IX (digital inputs)

Input type	24V, typ. 7mA, sink or source (PNP or NPN), opto isolated
Debounce	10ms software filter, common for all inputs
Update time	10ms (IEX-2 mode)

QX (digital outputs)

Output type	N-channel V-FET (current source, load to ground), opto isolated
Load	1A/30V DC
Update time	100ms (IEX-2 mode)

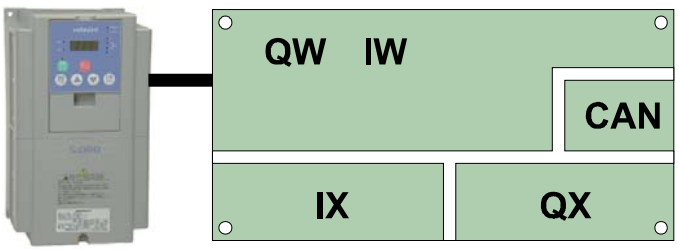
IW (analog inputs)

Input type	0..10V
Input resistance	10kohm
A/D converter	8-bit succesive approximation register
Conversion time	100us
Update time	100ms (IEX-2 mode)
Accuracy	2% of FSR at 25°C

QW (analog output)

Output type	0..10V
Output current	max. 10mA
D/A converter	8-bit resistor-string DAC, guaranteed monotonic
Settling time	100us
Update time	100ms (IEX-2 mode)
Accuracy	2% of FSR at 25°C

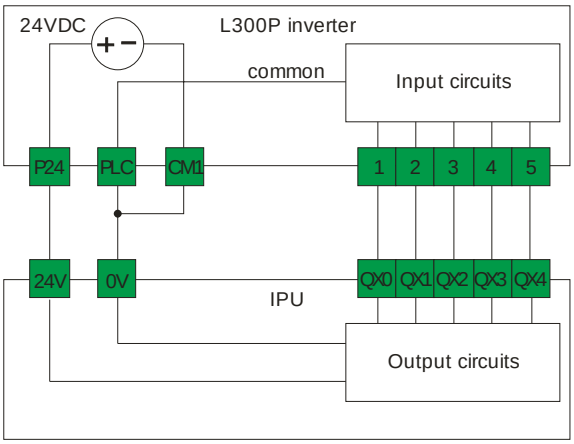
Galvanic isolation



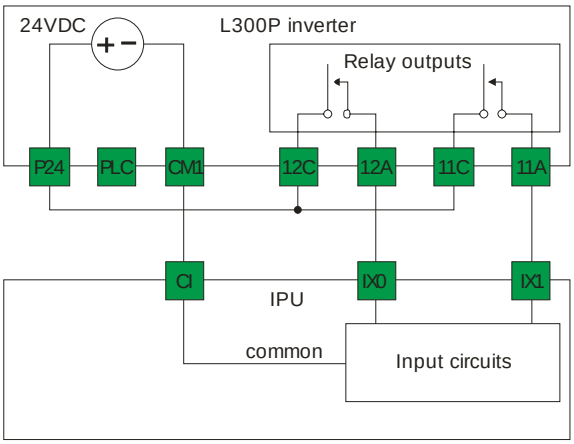
CAN bus is galvanically separated from CPU and main inverter body, but it is connected to inverter 24V power output (common to inverter digital inputs).

Wiring examples

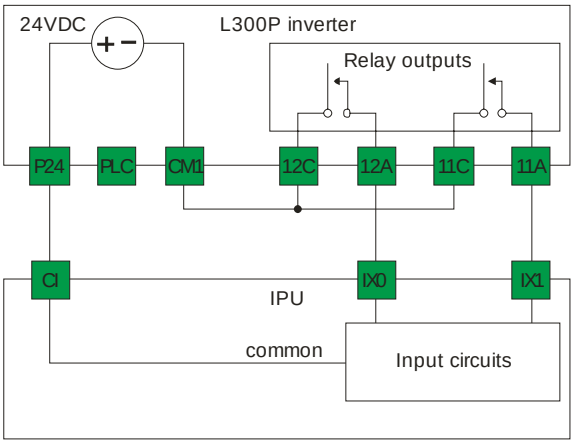
Connect IPU outputs to inverter inputs (common minus):



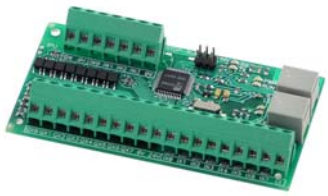
Connect inverter outputs to IPU inputs (common minus):



Connect inverter outputs to IPU inputs (common plus):



Order options

IPU-Pxx-EM 	IPU module with preloaded software: IPU-P00-EM - IEX-2 mode system software IPU-P01-EM - PLC mode multi-speed demo (source available)
OP-2i 	Operator panel for IPU.
ISP ADAPTER 	Kit for loading predefined programs to IPU. It contains: - ISP adapter - FLIP software PC with MS Windows 2000 (or later) and COM port is needed.
IPU STARTER PACK	Kit for writing new PLC-mode applications. It contains: - IPU-P01-EM module - OP-2i operator panel - 2m cable - PLC mode template and demo (C/asm source code) - ISP adapter - FLIP software - cupon for a free one-day training, valid for one person Kit does not include programming software. Any C51 compiler may be used, but for full compatibility with extensions used in template, KEIL C is suggested (www.keil.com).

Appendix

Address range:

Memory area		IPU address	Inverter address
External RAM	EEPROM buffer area	0000 - 03FF	0800 0000 - 0800 03FF
	RAM monitor area	0400 - 3FF0	0800 0400 - 0800 3FF0
Internal RAM	Inverter status area	4000 - 4FFF	FFFF 8000 - FFFF 8FFF

Short overview of frequently used SJ300/L300P parameters:

Identifier	Address	Size	Decimal places	Range	Default	Unit	Access	Name
Control								
A020	$0000	4	2	0..40000	0	Hz	Read/write	Multispeed Frequency 0
RunStop	$0472	1	0	0, 1, 2, 4	0		Read/write	Run command for operater
Monitoring								
D001	$0400	4	3	0..400000		Hz	Read	Output frequency
D002	$042C	2	1	0..10000		A	Read	Output current
D003	$044A	1	0	0..2			Read	Rotating direction
D004	$0404	4	2	0..999900		%	Read	PID feedback data
D005	$042E	2	0	0..65535			Read	Condition of input terminal
D006	$0446	2	0	0..65535			Read	Condition of output terminal
D007	$0408	2	0	0..65535			Read	Digital Input 1
D013	$0448	2	1	0..10000		V	Read	Output voltage
D014	$0432	2	1	0..10000		kW	Read	Input electrical power
D016	$0410	4	0	$0..2^{32}-1$		s	Read	RUN Time
D017	$0414	4	0	$0..2^{32}-1$		s	Read	Power ON Time
TripCnt	$0360	2	0				Read	
Configuration (standard)								
A001	$023F	1	1	0..5	1		Read/write[2]	Frequency source
A002	$0240	1	1	1..5	1		Read/write[2]	Run command source
A021	$000C	4	2	0..40000	0	Hz	Read/write	Multispeed frequency 1
A022	$0010	4	2	0..40000	0	Hz	Read/write	Multispeed frequency 2
A023	$0014	4	2	0..40000	0	Hz	Read/write	Multispeed frequency 3
A071	$0260	1	0	0..1	0		Read/write	PID Enable
C001	$0299	1	0	1..255	18		Read/write	Function of input terminal 1
C002	$029A	1	0	1..255	16		Read/write	Function of input terminal 2
C011	$02A1	1	0	0..1	0		Read/write	Condition of input terminal 1
C012	$02A2	1	0	0..1	0		Read/write	Condition of input terminal 2
F002	$0058	4	2	1..360000	3000	s	Read/write	Acceleration time
F003	$0064	4	2	1..360000	3000	s	Read/write	Deceleration time
Configuration (actionprocess[1])								
A004	$00F8	2	0	30..400	50	Hz	Read/write[2]	Maximum frequency
EE store	$044B	1	0	0..15	0		Read/write[2]	EEPROM store flag

Note [1]: Actionprocess parameters become active after reset.
Note [2]: Writing is possible only when inverter is stopped. Reading is not limited.